Converting Nondeterministic Automata and Context-Free Grammars into Parikh Equivalent Deterministic Automata

Giovanna J. Lavado[1] Giovanni Pighizzini[1] Shinnosuke Seki[2]

[1]Dipartimento di Informatica
Università degli Studi di Milano, Italy

[2]Department of Information and Computer Science
Aalto University, Finland

DLT 2012
台北、台湾
August 14–17, 2012

NFAs vs DFAs

Subset construction: [Rabin&Scott '59]

NFA	$\Longrightarrow$	DFA
n states		2^n states

The state bound cannot be reduced

[Lupanov '63, Meyer&Fischer '71, Moore '71]

What happens if we do not care of the order of symbols in the strings?

This problem is related to the concept of *Parikh Equivalence*

NFAs vs DFAs

Subset construction: [Rabin&Scott '59]

NFA	$\Longrightarrow$	DFA
n states		2^n states

The state bound cannot be reduced

[Lupanov '63, Meyer&Fischer '71, Moore '71]

What happens if we do not care of the order of symbols in the strings?

This problem is related to the concept of *Parikh Equivalence*

Parikh Equivalence

- $\Sigma = \{a_1, \ldots, a_m\}$ alphabet of m symbols
- *Parikh's map* $\psi : \Sigma^* \to \mathbb{N}^m$:

$$\psi(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_m})$$

  for each string $w \in \Sigma^*$
- *Parikh's image* of a language $L \subseteq \Sigma^*$:

$$\psi(L) = \{\psi(w) \mid w \in L\}$$

- $w' =_\pi w''$ iff $\psi(w') = \psi(w'')$
- $L' =_\pi L''$ iff $\psi(L') = \psi(L'')$

Parikh's Theorem

Theorem ([Parikh '66])

The Parikh image of a context-free language is a semilinear set, i.e, each context-free language is Parikh equivalent to a regular language

Example:

- $L = \{a^n b^n \mid n \geq 0\}$
- $R = (ab)^*$

$$\psi(L) = \psi(R) = \{(n, n) \mid n \geq 0\}$$

Different proofs after the original one of Parikh, e.g.

- [Goldstine '77]: a simplified proof
- [Aceto&Ésik&Ingólfsdóttir '02]: an equational proof
- ...
- [Esparza&Ganty&Kiefer&Luttenberger '11]: complexity aspects

Parikh's Theorem

Theorem ([Parikh '66])

The Parikh image of a context-free language is a semilinear set, i.e, each context-free language is Parikh equivalent to a regular language

Example:

- $L = \{a^n b^n \mid n \geq 0\}$
- $R = (ab)^*$

$$\psi(L) = \psi(R) = \{(n, n) \mid n \geq 0\}$$

Different proofs after the original one of Parikh, e.g.

- [Goldstine '77]: a simplified proof
- [Aceto&Ésik&Ingólfsdóttir '02]: an equational proof
- ...
- [Esparza&Ganty&Kiefer&Luttenberger '11]: complexity aspects

Parikh's Theorem

Theorem ([Parikh '66])

The Parikh image of a context-free language is a semilinear set, i.e, each context-free language is Parikh equivalent to a regular language

Example:

- $L = \{a^n b^n \mid n \geq 0\}$
- $R = (ab)^*$

$$\psi(L) = \psi(R) = \{(n, n) \mid n \geq 0\}$$

Different proofs after the original one of Parikh, e.g.

- [Goldstine '77]: a simplified proof
- [Aceto&Ésik&Ingólfsdóttir '02]: an equational proof
- ...
- [Esparza&Ganty&Kiefer&Luttenberger '11]: complexity aspects

Our Goal

We want to convert nondeterministic automata and context-free grammars into *small Parikh equivalent* deterministic automata

Problem (NFAs to DFAs)

NFA *n states*	$\Longrightarrow_\pi$	*DFA* *how many states?*

Problem (CFGs to DFAs)

CFG *size n*	$\Longrightarrow_\pi$	*DFA* *how many states?*

Why?

- Interesting theoretical properties:
  wrt Parikh equivalence regular and context-free languages are indistinguishable [Parikh '66]
- Connections of with:
  - Semilinear sets
  - Presburger Arithmetics [Ginsburg&Spanier '66]
  - Petri Nets [Esparza '97]
  - Logical formulas [Verma&Seidl&Schwentick '05]
  - Formal verification
    [Dang&Ibarra&Bultan&Kemmerer&Su'00, Göller&Mayr&To'09]
  - ...
- Unary case:
  size costs of the simulations of CFGs and PDAs by DFAs
  [Pighizzini&Shallit&Wang '02]

Why?

- Interesting theoretical properties:
  wrt Parikh equivalence regular and context-free languages are indistinguishable [Parikh '66]
- Connections of with:
  - Semilinear sets
  - Presburger Arithmetics [Ginsburg&Spanier '66]
  - Petri Nets [Esparza '97]
  - Logical formulas [Verma&Seidl&Schwentick '05]
  - Formal verification
    [Dang&Ibarra&Bultan&Kemmerer&Su'00, Göller&Mayr&To'09]
  - ...
- Unary case:
  size costs of the simulations of CFGs and PDAs by DFAs
  [Pighizzini&Shallit&Wang '02]

Why?

- Interesting theoretical properties:
  wrt Parikh equivalence regular and context-free languages are indistinguishable [Parikh '66]
- Connections of with:
  - Semilinear sets
  - Presburger Arithmetics [Ginsburg&Spanier '66]
  - Petri Nets [Esparza '97]
  - Logical formulas [Verma&Seidl&Schwentick '05]
  - Formal verification
    [Dang&Ibarra&Bultan&Kemmerer&Su'00, Göller&Mayr&To'09]
  - ...
- Unary case:
  size costs of the simulations of CFGs and PDAs by DFAs
  [Pighizzini&Shallit&Wang '02]

Converting NFAs

Problem (NFAs to DFAs)

NFA *n states*	$\Longrightarrow_\pi$	*DFA* *how many states?*

- Upper bound: 2^n (subset construction)
- Lower bound: $e^{\sqrt{n \ln n}}$
  This bound derives from the *unary case*:
  the state cost of the conversion of unary n-state NFAs into equivalent DFAs is $e^{\Theta(\sqrt{n \ln n})}$ [Chrobak '86]

Converting NFAs

Problem (NFAs to DFAs)

NFA *n states*	$\Longrightarrow_{\pi}$	*DFA* *how many states?*

- Upper bound: 2^n (subset construction)
- Lower bound: $e^{\sqrt{n \ln n}}$
  This bound derives from the *unary case*:
  the state cost of the conversion of unary n-state NFAs
  into equivalent DFAs is $e^{\Theta(\sqrt{n \ln n})}$ [Chrobak '86]

Converting NFAs: General Idea

A	n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

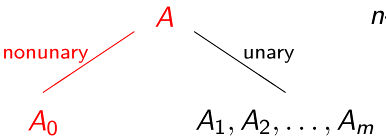
Converting NFAs: General Idea

A $\quad$ n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

unary

$A_1, A_2, \ldots, A_m$ $\quad$ $L(A_i) = L(A) \cap a_i^*,\ i \geq 1$

Converting NFAs: General Idea



n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

$L(A_i) = L(A) \cap a_i^*$, $i \geq 1$

$L(A_0) = L - \bigcup_{i=0}^{m} L(A_i)$

Converting NFAs: General Idea

A

nonunary | unary

A_0 | $A_1, A_2, \ldots, A_m$

A'_0 | $A'_1, A'_2 \ldots, A'_m$

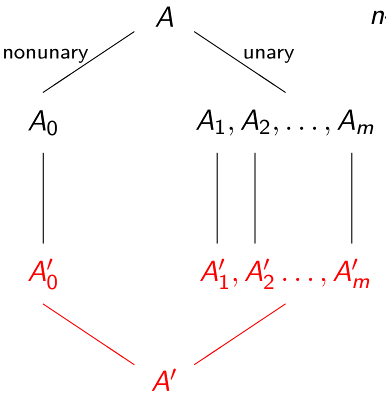
n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

$L(A_i) = L(A) \cap a_i^*,\ i \geq 1$

$L(A_0) = L - \bigcup_{i=0}^{m} L(A_i)$

Parikh equivalent DFAs

Converting NFAs: General Idea



n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

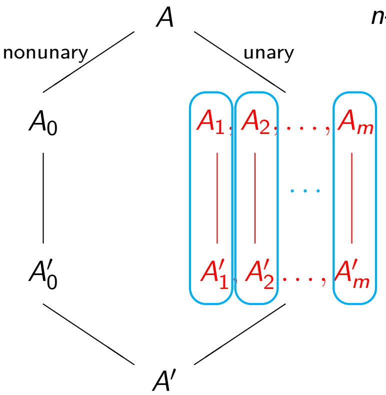
$$L(A_i) = L(A) \cap a_i^*, \; i \geq 1$$

$$L(A_0) = L - \bigcup_{i=0}^{m} L(A_i)$$

Parikh equivalent DFAs

DFA Parikh equivalent to A

Converting NFAs: General Idea



n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

$L(A_i) = L(A) \cap a_i^*,\ i \geq 1$

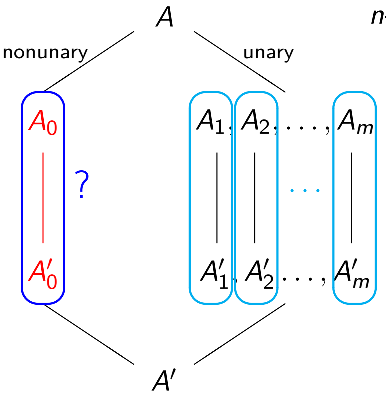
$L(A_0) = L - \bigcup_{i=0}^{m} L(A_i)$

Chrobak conversion:
$e^{O(\sqrt{n \ln n})}$ states

Parikh equivalent DFAs

DFA Parikh equivalent to A

Converting NFAs: General Idea



n-state NFA over $\Sigma = \{a_1, \ldots, a_m\}$

$L(A_i) = L(A) \cap a_i^*,\ i \geq 1$

$L(A_0) = L - \bigcup_{i=0}^{m} L(A_i)$

Chrobak conversion:
$e^{O(\sqrt{n \ln n})}$ states

Parikh equivalent DFAs

DFA Parikh equivalent to A

How much it costs the conversion of NFAs accepting *only nonunary strings* into Parikh equivalent DFAs?

Converting NFAs Accepting Only Nonunary Strings

Problem (NFAs to DFAs, restricted)

NFA s.t. each accepted string is nonunary *n states*	$\Longrightarrow_\pi$	*DFA* *how many states?*

Quite surprisingly, we can obtain a DFA with a number of states *polynomial* in n,

i.e., this conversion is less expensive than the conversion in the unary case, which costs $e^{\Theta(\sqrt{n \ln n})}$

Converting NFAs Accepting Only Nonunary Strings

The conversion uses a modification of the following result:

Theorem ([Kopczyński&To '10])

Given $\Sigma = \{a_1, \ldots, a_m\}$, there is a polynomial p s.t. for each n-state NFA A over Σ,

$$\psi(L(A)) = \bigcup_{i \in I} Z_i$$

where:

- *I is a set of at most $p(n)$ indices*
- *for $i \in I$, $Z_i \subseteq \mathbb{N}^m$ is a linear set of the form:*

  $$Z_i = \{\alpha_0 + n_1\alpha_1 + \cdots + n_k\alpha_k \mid n_1, \ldots, n_k \in \mathbb{N}\}$$

  with
  - $0 \leq k \leq m$
  - *the components of α_0 are bounded by $p(n)$*
  - *$\alpha_1, \ldots, \alpha_k$ are linearly independent vectors from $\{0, 1, \ldots, n\}^m$*

Converting NFAs Accepting Only Nonunary Strings

Outline: linear sets

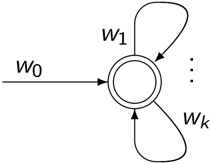
Each above linear set

$$Z_i = \{\alpha_0 + n_1\alpha_1 + \cdots + n_k\alpha_k \mid n_1, \ldots, n_k \in \mathbb{N}\}$$

can be converted into a poly size DFA accepting a language

$$R_i = w_0(w_1 + \cdots + w_k)^*$$

s.t. $\psi(w_j) = \alpha_j$, $j = 0, \ldots, k$, and
$w_1, \ldots, w_k$ *begin with different letters*



Each above linear set

$$Z_i = \{\alpha_0 + n_1\alpha_1 + \cdots + n_k\alpha_k \mid n_1, \ldots, n_k \in \mathbb{N}\}$$

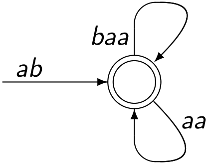
can be converted into a poly size DFA accepting a language

$$R_i = w_0(w_1 + \cdots + w_k)^*$$

s.t. $\psi(w_j) = \alpha_j$, $j = 0, \ldots, k$, and
$w_1, \ldots, w_k$ *begin with different letters*

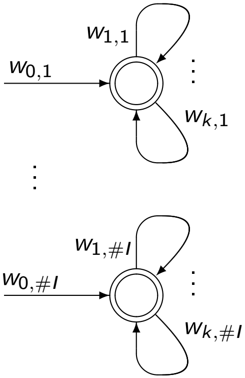
Example:

- $\{(1,1) + n_1(2,1) + n_2(2,0) \mid n_1, n_2 \geq 0\}$
- $ab(baa + aa)^*$



Converting NFAs Accepting Only Nonunary Strings

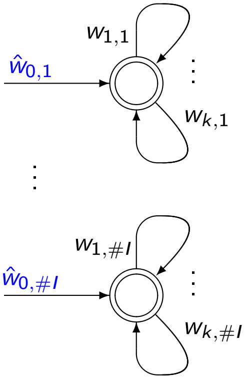
Outline: from linear to semilinear



- Standard construction for union of DFAs: number of states = *product*
  $\#I \leq p(n) \;\Rightarrow$ Too large!!!
- Strings $w_{0,i}$ can be replaced by Parikh equivalent strings $\hat{w}_{0,i}$ in such a way that $W_0 = \{\hat{w}_{0,i} \mid i \in I\}$ is a *prefix code*
- After this change:
  number of states $\leq$ *sum* Polynomial!!!

Converting NFAs Accepting Only Nonunary Strings

Outline: from linear to semilinear



- Standard construction for union of DFAs: number of states $=$ *product*

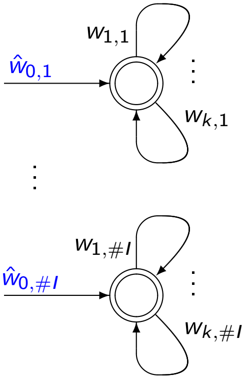
  $\#I \leq p(n) \;\Rightarrow$ Too large!!!

- Strings $w_{0,i}$ can be replaced by Parikh equivalent strings $\hat{w}_{0,i}$ in such a way that $W_0 = \{\hat{w}_{0,i} \mid i \in I\}$ is a *prefix code*

- After this change: number of states $\leq$ *sum* Polynomial!!!

Converting NFAs Accepting Only Nonunary Strings

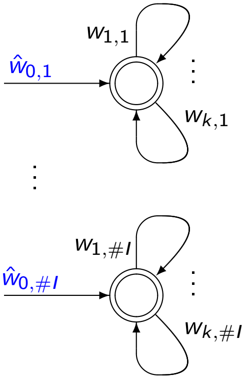
Outline: from linear to semilinear



- Standard construction for union of DFAs: number of states = *product*
  $\#I \leq p(n) \ \Rightarrow$ Too large!!!
- Strings $w_{0,i}$ can be replaced by Parikh equivalent strings $\hat{w}_{0,i}$ in such a way that $W_0 = \{\hat{w}_{0,i} \mid i \in I\}$ is a *prefix code*
- After this change:
  number of states $\leq$ *sum* Polynomial!!!

Converting NFAs Accepting Only Nonunary Strings

Outline: from linear to semilinear

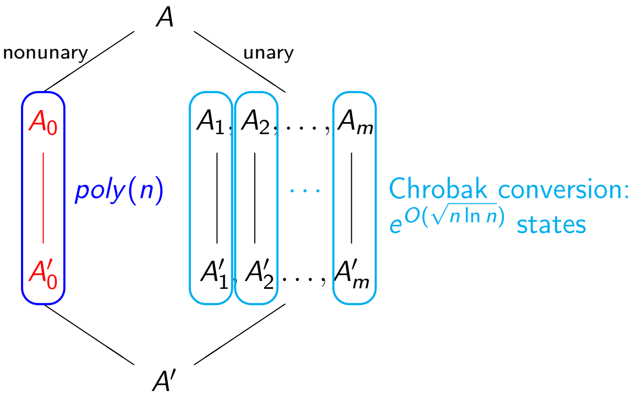


- Standard construction for union of DFAs:
  number of states $=$ *product*
  $\#I \leq p(n) \;\Rightarrow$ Too large!!!
- Strings $w_{0,i}$ can be replaced by Parikh equivalent strings $\hat{w}_{0,i}$ in such a way that $W_0 = \{\hat{w}_{0,i} \mid i \in I\}$ is a *prefix code*
- After this change:
  number of states $\leq$ *sum* Polynomial!!!

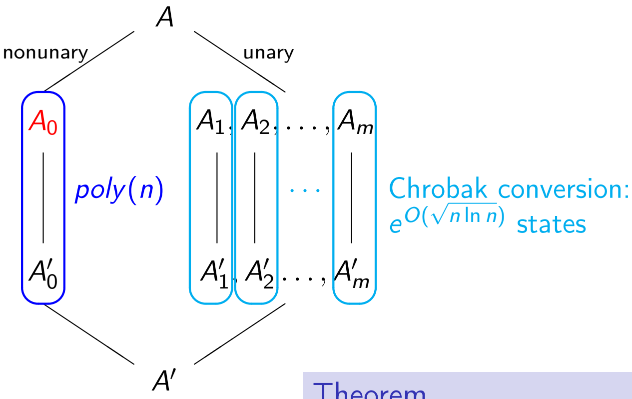
Theorem

For each n-state NFA accepting a language none of whose words are unary, there exists a Parikh equivalent DFA with a number of states polynomial in n

Converting NFAs: Back to the General Case



Converting NFAs: Back to the General Case



Theorem

For each n-state NFA there exists a Parikh equivalent DFA with $e^{O(\sqrt{n \ln n})}$ states. Furthermore, this cost is tight

Converting CFGs

Problem (CFGs to NFAs and DFAs)

CFG		NFA/DFA
size h	$\Longrightarrow_\pi$	how many states?

- We consider CFGs in Chomsky Normal Form
- As a measure of size we consider the *number of variables*

[Gruska '73]

Problem (CFGs to NFAs)

CFG *Chomsky normal form* *h variables*	$\Longrightarrow_\pi$	*NFA* *how many states?*

Upper bound:

- $2^{2^{O(h^2)}}$ implicit construction from classical proof of Parikh's Th.
- $O(4^h)$ [Esparza&Ganty&Kiefer&Luttenberger '11]

Lower bound: $\Omega(2^h)$ Folklore

Problem (CFGs to DFAs)

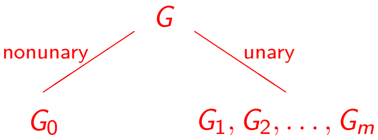
CFG *Chomsky normal form* *h variables*	$\Longrightarrow_\pi$	*DFA* *how many states?*

- Upper bound: $2^{O(4^h)}$ — subset construction
- Lower bound: 2^{ch^2} — tight bound for the unary case $2^{\Theta(h^2)}$ [Pighizzini&Shallit&Wang '02]

Converting CFGs into Parikh Equivalent DFAs

G CFG with h variables

Converting CFGs into Parikh Equivalent DFAs



CFG with h variables

Converting CFGs into Parikh Equivalent DFAs

G

nonunary — unary

G_0 — $G_1, G_2, \ldots, G_m$

4^h

A_0

$poly(4^h)$

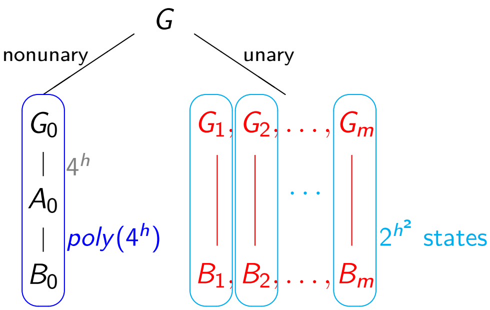
B_0

CFG with h variables

Parikh equivalent NFA

Parikh equivalent DFA

Converting CFGs into Parikh Equivalent DFAs

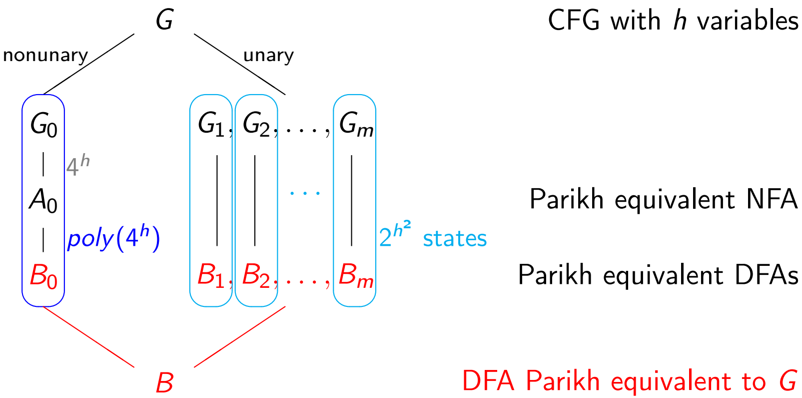


CFG with h variables

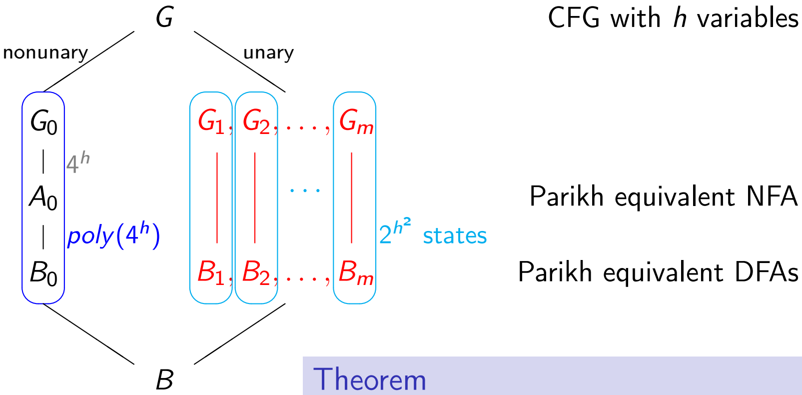
Parikh equivalent NFA

Parikh equivalent DFAs

Converting CFGs into Parikh Equivalent DFAs



Converting CFGs into Parikh Equivalent DFAs



Theorem

For any CFG in Chomsky normal form with h variables, there exists a Parikh equivalent DFA with at most $2^{O(h^2)}$ states. Futhermore this bound is tight

Final considerations

We obtained the following tight conversions:

NFA n states	$\Longrightarrow_\pi$	DFA $e^{O(\sqrt{n \ln n})}$ states
CFG Chomsky normal form h variables	$\Longrightarrow_\pi$	DFA $2^{O(h^2)}$ states

- In both cases the most expensive part is the unary one
- It could be interesting to investigate if for other constructions related to regular and context-free languages similar phenomena happen (e.g., automata minimization, state complexity of operations, ...)

Final considerations

We obtained the following tight conversions:

NFA n states	$\Longrightarrow_\pi$	DFA $e^{O(\sqrt{n \ln n})}$ states
CFG Chomsky normal form h variables	$\Longrightarrow_\pi$	DFA $2^{O(h^2)}$ states

- In both cases the most expensive part is the unary one
- It could be interesting to investigate if for other constructions related to regular and context-free languages similar phenomena happen (e.g., automata minimization, state complexity of operations, ...)

Thank you for your attention!